A REPRESENTATION SCHEME FOR RAPID 3-D COLLISON DETECTION

NAGW-1333

By:

R.B. Kelley

Department of Electrical, Computer and Systems Engineering
Department of Mechanical Engineering, Aeronautical Engineering & Mechanics
Rensselaer Polytechnic Institute
Troy, New York 12180-3590

CIRSSE Document #9

Presented: Third International Symposium on Intelligent Control, Aug. 1988.

A REPRESENTATION SCHEME FOR RAPID 3-D COLLISION DETECTION

SUSAN BONNER AND ROBERT B. KELLEY

Electrical, Computer, and Systems Engineering Department
Rensselaer Polytechnic Institute
Troy, New York 12180-3590

Abstract **— A new scheme is presented for the representation of objects for the rapid detection of collisions in a dynamic three-dimensional environment such as when a proposed path for a robot-carried object is tested for collisions with obstacles along the path. The successive spherical approximation (SSA) representation provides a representation scheme which allows rapid collision detection while still providing for the exact representation of dynamic objects. The hierarchy of representation levels is based on the subdivision of a sphere. The degree of approximation of the SSA representation decreases as the scheme traverses down the representation tree. The creation of the SSA representation is illustrated using an object database containing boundary representations. The use of the SSA hierarchical representation for the rapid and exact collision detection between three-dimensional objects and the environment is shown by considering the collision detection problem between stationary and moving objects. It is shown that there is little additional cost incurred by allowing changes in position and orientation of the objects. The contribution of the SSA representation is that it leads to a time-efficient, hybrid collision detection scheme.**

I. INTRODUCTION

Collision detection is a subset of the robot path planning problem in which a given path for an object through the robot workspace is tested for collisions with the obstacles in the workspace. Many representations of the environment have been used to facilitate the collision detection process. These include sweep representations, constructive solid geometry, boundary representations, cell decompositions and spatial occupancy enumerations [Hayward, 1]. Of these, a form of spatial occupancy enumeration, in which the robot environment is divided into a hierarchy of *mixed*, *full*, and *empty* nodes has been most popular. In these methods, the environment is mapped into the joint space of the robot, allowing a path to be determined by maneuvering a point through the hierarchical model of the modified space [Lozano-Perez, 2]. Octrees have been found to be an effective method for this type of solution in 3-D [Faverjon, 3], although other representations have also been used [Brooks, 4; Gouzenes, 5]. There are two fundamental problems with these methods. The first is that the computational complexity required to find a path is extremely high, resulting in several hours of computing time for a single path. This is magnified by the need to recreate the space in the event of a change in the environment, a frequent occurrence in a dynamic robot workcell. The second problem is in the loss of intrinsic information caused by the conversion from the operational space of the robot to the joint space. Computation of the shortest path is computationally impractical and the use of heuristic reasoning to guide path planning decisions is severely limited.

Attempts to perform collision detection in operational space have been hindered by the lack of a good representation. There are two types of collision detection required for path planning in operational space: the swept volume method and the incremental method [Leu, 6]. The swept volume method examines the intersection of the volume swept by the object over the path with the volumes of the obstacles [Pennington with others, 7].

This method is hindered by the difficulty involved in creating the three dimensional swept volume of the object and in the limitations placed on the allowable orientation changes of the object over the path. The incremental method performs collision detection at incremental locations along a path, thus allowing orientation change [Meyers and Agin, 8]. However, the underlying difficulty with this method is in the determination of increments which minimize the computational effort required to test multiple points of intersection and still guarantee a safe path. Incremental collision

detection can be expanded to provide path planning via the use of a generate-and-test philosophy in which a part of the path can be altered to avoid a collision. Another form of path planning in operational space involves finding a path through obstacles by assigning potential fields to the objects and then finding a path which minimizes the degree to which they repel each other [Khatib, 9]. A variation on this technique uses sensing in the robot to repel obstacles as it makes its way through the robot environment [Lumelsky, 10]. These methods provide a rapid means of finding a path in an uncluttered environment, but they are hindered by an inexactness in the representation of the objects, a tendency toward deadlock and cycling, and no guarantee of a reasonable path.

The successive spherical approximation (SSA) representation provides a representation scheme which allows rapid collision detection, while still providing an exact representation of dynamic objects. The representation can be used to perform both swept volume and incremental collision detection, but the hierarchical nature of the representation favors a hybrid detection process which uses elements from both approaches. In addition, SSA allows the exploitation of human three-dimensional conceptual knowledge. This makes the SSA representation ideal for use in a generate and test path planning situation.

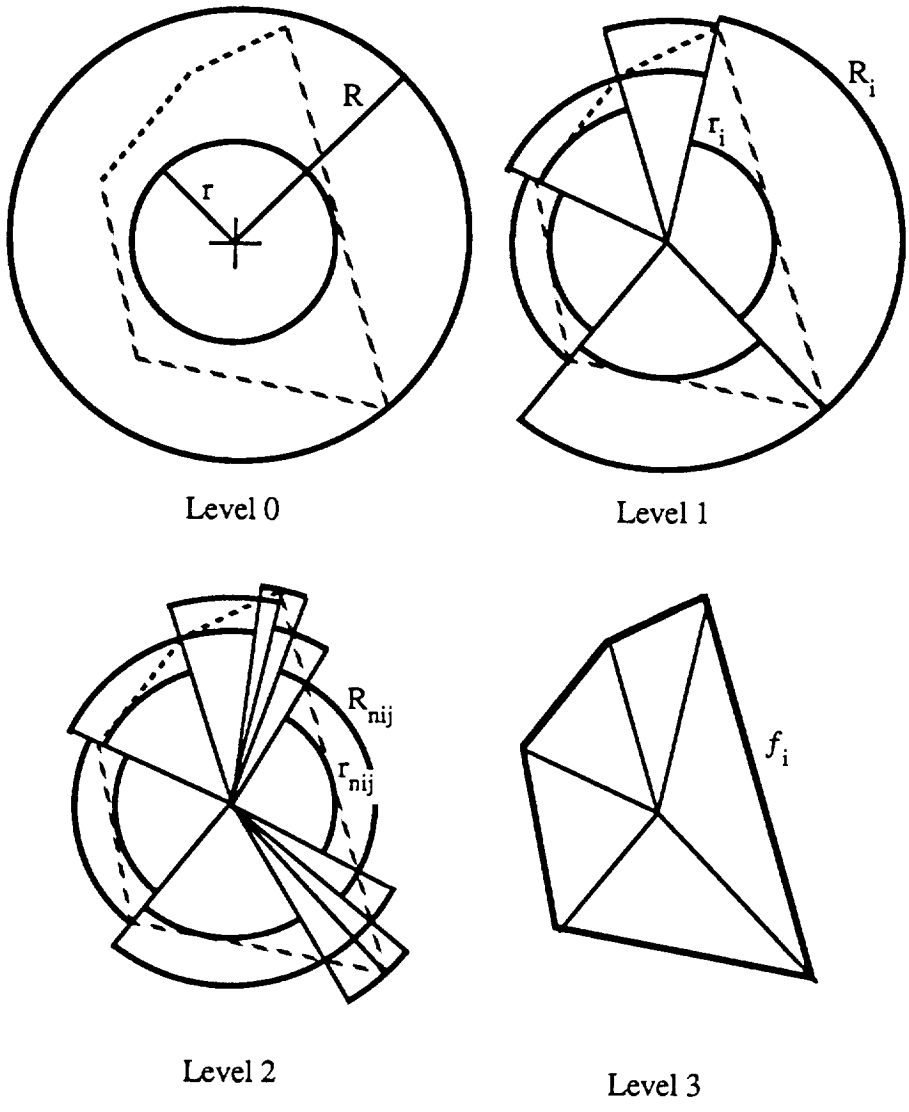


Fig. 1 SSA representation hierarchy of a convex planar object. (2-D depiction)

An SSA representation of an object is made up of a hierarchy of representation levels based on the division of an object encircling sphere into spherical sectors, as depicted in Figure 1. The most coarse approximation is a pair of spheres with a common center. The outer sphere contains the object and the inner sphere is contained within the object. The next finer approximation is formed by dividing the spheres into spherical sectors defined by the object faces. Each segment is assigned an inner and outer radius based on the geometry of the face. The third, most refined, approximation is formed by the further division of those spherical sectors which provide an inadequate representation of a face. The final level of the approximation hierarchy is composed of the planar surfaces of the faces of the object itself, an exact representation. As is illustrated in this paper, the hierarchical nature of the SSA representation levels allows for rapid and exact collision detection between three-dimensional objects in a robot workspace with little additional cost incurred by changes in position and orientation of the objects.

The following sections introduce the SSA representation hierarchy in detail by describing how the SSA is created from an object database containing boundary representations. Next we show how collision detection can be accomplished between stationary objects. Using these techniques, we then present a collision detection scheme for an object moving along a straight line path within a robot environment. Finally, we discuss the applicability of SSA to the generate-and-test path planning problem in a robot workcell environment.

II. Creating the SSA Representation

The following section details the creation of the representation from a CAD-like database containing boundary representations of convex planar objects. In addition, we assume the representation includes the location of the center of mass (centroid) of each of the objects with respect to some world coordinate system and the locations of the object vertices specified with respect to its centroid. We define object edges by a unit direction vector and a vertex and object faces by a unit normal vector and a vertex.

A. Level 0 : Object Bounding Spheres

The most approximate level involves the determination of outer and inner spheres bounding spheres having a common center. Although any center may be used, we use the location of the centroid because it is commonly available in object representations used for robotics applications. The radius of the outer sphere of a convex planar object is the maximum of the distances from the object center to the vertices of the planar faces. The radius of the inner sphere is the minimum of the minimum distances from the object center to the faces of the convex planar object.

B. Level 1: Facial Bounding Spheres

The spherical coordinates of each vertex, relative to the object center C, is denoted by (α, β, r). The facial spherical approximation assigns inner and outer radii to a spherical sector defined by the spherical coordinates of the vertices of a face. The rectangular scope of each facial spherical sector is determined by the range of α and β swept by the face as shown in Figure 2. The inner and outer radii of the spherical sectors are determined by the same method used to determine the radii of the bounding spheres for level 0, but only the distances for the desired face are used.

Note that the use of ranges does not give an exact angular sweep of the face but a rectangular shaped sector. However, since this is an approximation, the loss of precision caused by the use of the rectangular sectors is outweighed by the savings in computational complexity over a more precise representation. (Care must be taken when assigning ranges if the β-axis passes through or touches the face, or if α changes from $-\pi$ to $+\pi$.)

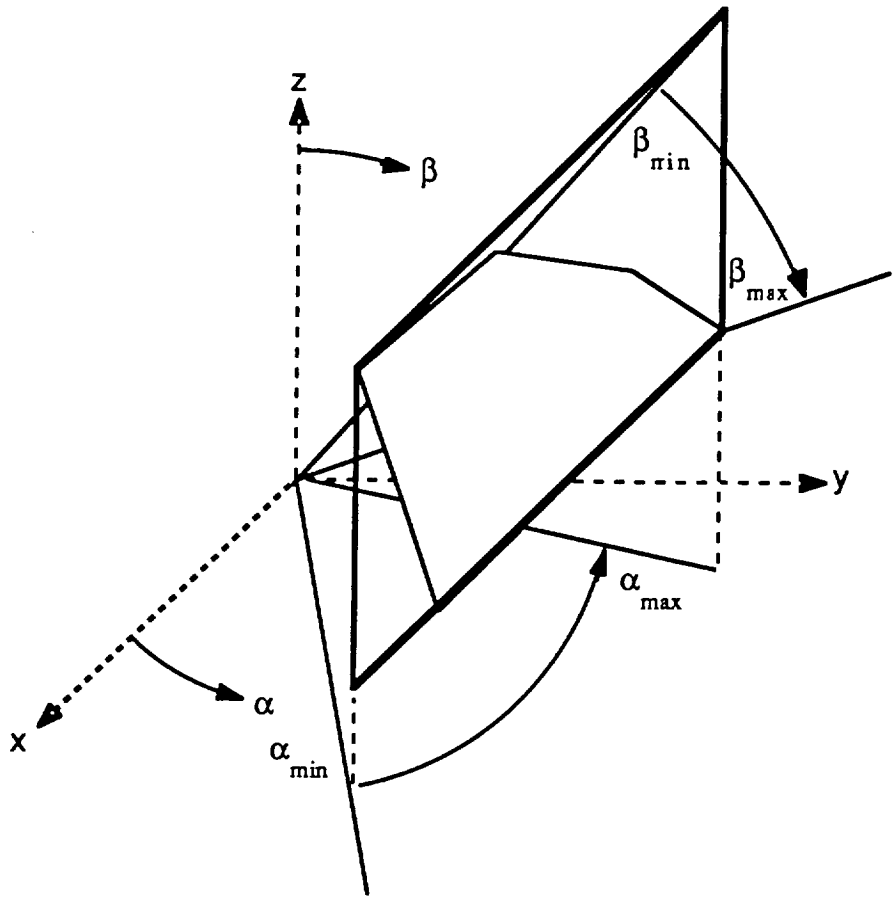


Fig. 2. Rectangular scope of facial sector: range of α and β.

C. Level 2: Approximate Subdivision of a Face

Depending upon the collision detection technique chosen and the geometry of the faces of the object, further division of some faces, especially large ones, may be desired. This further division may be done in many ways. To obtain an approximation which further refines the level 1 representation, we use the following technique (illustrated in Figure 3 for a typical planar face).

1. Define the base point, $B(\alpha_B, \beta_B, r_B)$, as the point where the face is closest to the object center, C. Let D_i be the distance from C to v_i. Define directed lines l_i which originate at B and pass through each of the vertices v_i on the face. Let *INC* be the radial increment and let n be the index of the increment.

Perform steps 2 through 4 to find S_{nij} until d_{ni} exceeds D_i, the distance to vertex v_i, for all i.

2. Find a point Q_{ni} on each of the lines l_i for which

$$Q_{ni} = (\alpha_{ni}, \beta_{ni}, d_{ni}) , \text{ where } d_{ni} = R_B + n\,INC .$$

3. For each pair of adjacent vertices v_i and v_j, define the subface by the four points $[P_{(n-1)i}, P_{(n-1)j}, P_{ni}, P_{nj}]$.

If $d_{ni} < D_i$ then $P_{ni} = Q_{ni}$. Otherwise, $P_{ni} = v_i$.

Set $(\alpha_{ni}, \beta_{ni}, r_{ni}) = P_{ni}$.

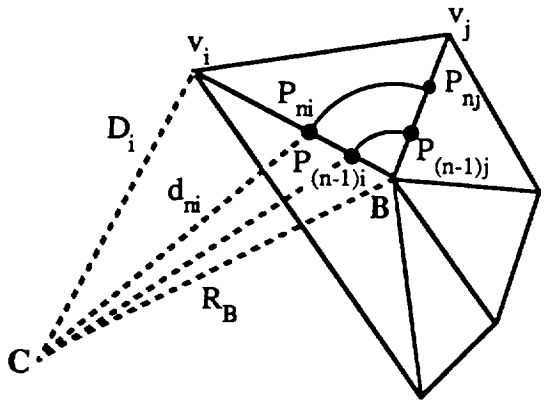
4. Find spherical sectors S_{nij} for each subface. The radii are given by

$$r_{nij} = min_{ij}\{r_{(n-1)i}, r_{(n-1)j}\} \text{ , and } R_{nij} = max_{ij}\{R_{ni}, R_{nj}\} .$$

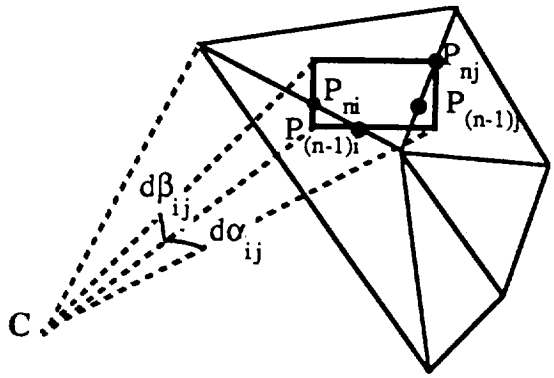
The ranges for these spherical sectors are

$$min\{\alpha_{(n-1)i}, \alpha_{(n-1)j}, \alpha_{ni}, \alpha_{nj}\} < \alpha_{ij} < max(\alpha_{(n-1)i}, \alpha_{(n-1)j}, \alpha_{ni}, \alpha_{nj}\} ,$$

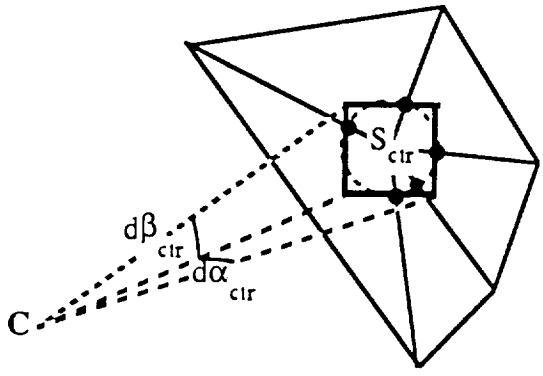
$$min\{\beta_{(n-1)i}, \beta_{(n-1)j}, \beta_{ni}, \beta_{nj}\} < \beta_{ij} < max\{\beta_{(n-1)i}, \beta_{(n-1)j}, \beta_{ni}, \beta_{nj}\} .$$



a) Steps 1 and 2



b) Steps 3 and 4



c) Step 5

Fig. 3. Approximate subdivision of a face.

5. Since S_{1ij} all intersect at a common point, B, they can be combined into one central rectangular sector, S_{ctr}, as follows:

$$r_{ctr} = r_B \text{ , and } R_{ctr} = r_B + INC \text{ ;}$$

then

$$min_{ij}\{\alpha_{1ij}\} < \alpha_{ctr} < max_{ij}\{\alpha_{1ij}\} \text{ ,}$$
$$min_{ij}\{\beta_{1ij}\} < \beta_{ctr} < max_{ij}\{\beta_{1ij}\} \text{ .}$$

This procedure creates a series of spherical sectors which more closely covers the surface of the object than do the facial bounding spheres. Note that each of the spherical rectangles overlaps each other. Within each overlap, the approximation by one spherical rectangle may be poor, but not by both. Thus, the desired covering is achieved. This is an example of our decision to give up some precision in the approximation to reduce the computational complexity of the tests. The degree of approximation depends upon the method of collision detection employed as discussed in the next section.

D. Level 3: Faces

If the above approximation levels fail to confirm a suspected collision, they do identify the specific faces of the objects which may be involved in a collision. If the intersection of the planes containing these faces intersect within the bounds of the faces, then there is a collision.

These SSA levels can now be used to find collisions between the objects. Note that the above procedure is done only once prior to invoking the collision detection process. The basic structure of the SSA representation does not change as the object moves throughout the workspace since the SSA representation is defined relative to the centroid of each object. Updates to the angle ranges and face definitions due to motion of the objects in the workspace are performed only when they are required to detect collisions. Because of the angle-based spherical division, changes in orientation are handled directly. Also, the SSA can potentially be used to represent objects of more complexity than convex planar objects without changing its basic structure or use.

III. COLLISIONS

Once the SSA representation is created, collision detection proceeds for the objects in the workspace. First, we consider collisions between two stationary objects and then we treat collisions when one object is moving along a straight line path.

A. Point Collisions

Point collisions can occur between two stationary objects modeled using SSA techniques. A collision is determined by comparing the overlap between each level of the representation. Point collisions are used by incremental collision detection strategies to ascertain an exact collision at a specific point in time between moving objects.

Determining point collisions using SSA techniques requires the computation of the base line d_B, the line between the centers of the two objects. The length of this line determines the distance between the objects and the degree of overlap at each level. The direction of the base line from the center of each object determines which part of the SSA representation is potentially involved in a collision.

At the bounding sphere level, level 0, the length of the base line, d_B, is all that is required to determine one of the three possible outcomes which occur at all levels. As illustrated in Figure 4, if d_B is greater than the sum of the radii of the outer spheres, there is NO COLLISION. If d_B is less than the sum of the inner spheres, there is a COLLISION. If the distance is somewhere between the two sums, the result is UNKNOWN, and the next level must be used to determine a collision.

At the facial level, level 1, the faces to check are determined by the spherical location of the base line with respect to the coordinate systems of each of the objects, the length of the base line, and the outer radii of the two bounding spheres. Figure 5 shows the relationships between these values in determining the kite-shaped area of potential collisions between two objects. We call this area a *kite* and use geometry to determine its parameters. Once the parameters have been established, facial spherical sectors which overlap the kite are found, and their radii compared to determine collisions.

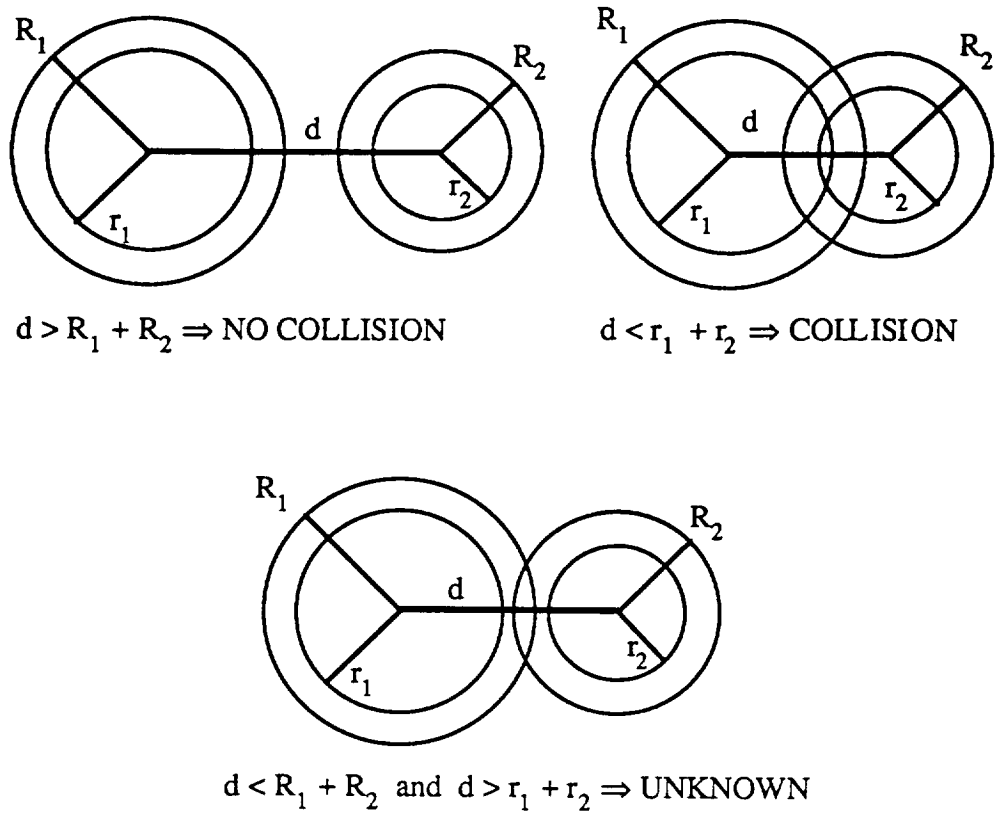


Fig. 4. Static collision - level 0, bounding sphere level possibilities.

To minimize the computational effort needed to get all the collision information from the kite, the orientation of the two spherical representations of the faces must coincide. This allows values of α and β to be transferred easily from one side of the kite to the other. This is accomplished simply by requiring the spherical coordinate system for each object to be aligned with a common world spherical coordinate system. When the orientation of an object changes, only the sweep ranges of α and β must be updated. This is done by updating the spherical coordinates of each object vertex with respect to the center and recalculating the ranges from the vertices of each face. Note that any vertices involved in checking collisions between the planar faces must be updated to recalculate the face definitions.

Once the orientations of the objects have been aligned, the kite between the two objects is established by defining a base range for each object. To facilitate comparisons, the range is defined in terms of the offset δ from the base angles. The base range also includes the outer radius of the object sphere, and the projection of the radius *n* onto the base line. Let the cartesian representation of $C_1 = (x_1, y_1, z_1)$ and $C_2 = (x_2, y_2, z_2)$.Then the base parameters are determined as follows:

$$\alpha_{1base} = \arctan\left(\frac{y_2 - y_1}{x_2 - x_1}\right),$$

$$\beta_{1base} = \arctan\left(\frac{\sqrt{(y_2 - y_1)^2 + (x_2 - x_1)^2}}{z_2 - z_1}\right),$$

$$\delta_{1base} = \arccos\left(\frac{R_1^2 + d_B^2 - R_2^2}{2R_1 d_B}\right),$$

$$n_{1base} = d_B - R_2 \cos(\delta_{2base}).$$

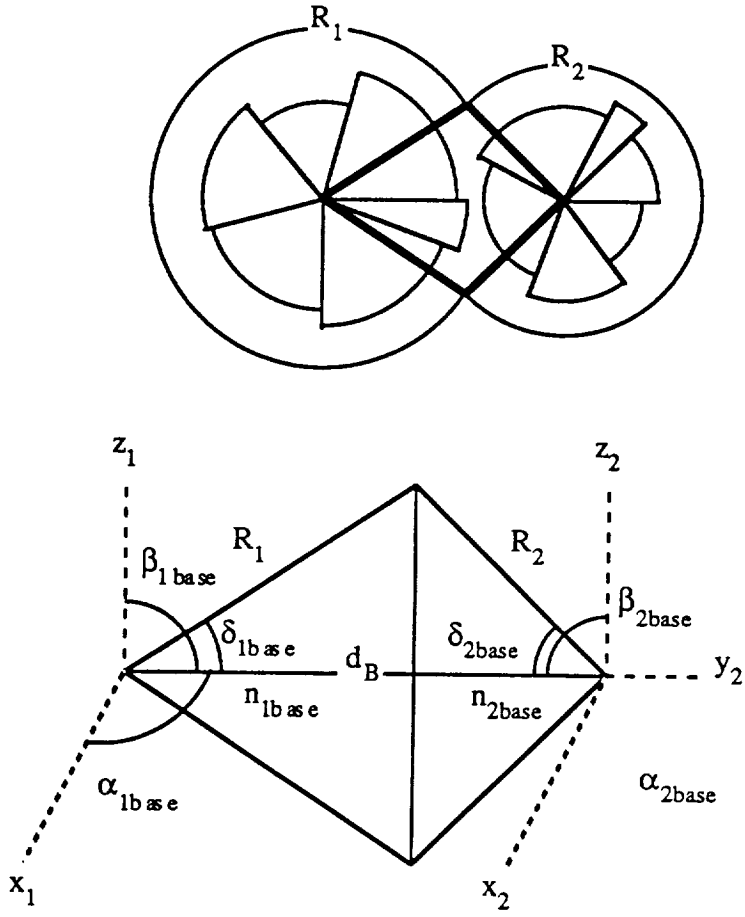


Fig. 5. Static collisions - level 1, facial bounding sphere level: definition of *kite* and its parameters

The next step is to find the spherical sectors in the objects which intersect the base range. This is done by redefining the angles swept by each spherical sector in terms of the base line and a pair of offsets, δ_{1j} and δ_{2j}, and determining whether the offsets overlap the base range. However, in order to be consistent for comparison purposes, the difference may need to be adjusted by $\pm 2\pi$. A min-max technique is used to determine whether the newly defined ranges intersect the base range:

$$\delta min_j = min\{\delta_{1j}, \delta_{2j}, \delta_{base}, -\delta_{base}\} ,$$
$$\delta max_j = max[\{\delta_{1j}, \delta_{2j}, \delta_{base}, -\delta_{base}\} ;$$
$$\text{if } (\delta min_j > \delta max_j) \text{ then NO OVERLAP .}$$

If there is an intersection, then δmin_i and δmax_j give the range for the overlapping spherical sector. Note that δ_1 and δ_2 must be determined and the overlap found for both α and β in order for an intersection to occur. Once an intersecting range has been determined, the corresponding range on the other object must be calculated and the distances compared. The offset angles are transferred across the kite using simple trigonometry. When α's are transferred, the range must be negated and reversed. β transfers are straight forward unless the β-axis passes through the face, in which case β ranges are negated and reversed.

The transferred range is intersected with the spherical sector ranges within the kite to determine the overlapping range, in the same manner as the spherical sectors were intersected with the base range. A quick check for a collision is made at this point using the outer radii (R_1 and R_2) of the two spherical sectors involved in the potential collision; if $(R_1 + R_2 < d_B)$ then NO COLLISION.

If this fails, we have a list of spherical sector pairs which we now compare to determine a collision. Although the ranges are all within the kite, they do not necessarily contain the base line and therefore, cannot be compared on the basis of d_B. The ranges of α and β for each spherical sector define a range of distances where collision detection is undetermined. This range has a minimum *d* and a maximum *D* which is determined using the trigonometry of the kite:

$$\delta_d = min\{\delta\alpha_1, \delta\alpha_2, \delta\beta_1, \delta\beta_2\} , \text{ and } d = \frac{n_{base}}{\cos(\delta_d)} .$$

Using these distances $\{d_1, D_1, d_2, D_2\}$ and the inner and outer radii of the corresponding spherical sectors (r_1, R_1, r_2, R_2),we now determine whether a collision will occur:

$$\text{if } [(r_1 \geq D_1) \text{ and } (r_2 \geq D_2)] \text{ then COLLISION ;}$$
$$\text{if } [(R_1 < d_1) \text{ and } (R_2 < d_2)] \text{ then NO COLLISION .}$$

Otherwise, the result is UNKNOWN and the next level must be used to determine a collision.

At this point, the procedure described above could be used to compare the rectangular sectors of subfaces for the pairs of sectors for which facial collisions were UNKNOWN. However, in the case of planar surfaces, the computational expense of finding a collision between the pairs of planar faces (which are now known) is small enough that the comparison of subface-approximated spherical sectors is not warranted.

The final step is to compare the planar faces themselves and determine whether the defining planes overlap within the bounds of the faces. Now is the time to recalculate the definition for the required faces if they have changed position and orientation since the last collision check.

B. Line Collisions

Line collisions are used to determine collisions between objects when one is stationary and the other is moving through the environment along a straight line. SSA techniques determine collision-free paths between objects to within a given degree of approximation with a reasonable amount of speed, which naturally depends upon the approximation error allowed. They are used to easily identify spherical sectors along a path in which a moving object can alter its orientation freely. SSA line collisions also identify portions of a path which may require closer examination via incremental methods and identify the faces involved.

When an object is moving along a straight line path in the environment, determination of a collision begins by finding the point on the line Pc at which is closest to the object to be checked.

The distance between the objects at this point is d_{Pc}. As illustrated in Figure 6, if d_{Pc} exceeds the sum of the outer radii of the bounding spheres, then there is NO COLLISION for the entire path. If d_{Pc} is less than the sum of the inner radii, then there is a COLLISION for the path. If d_{Pc} is between these two sums, the result is UNKNOWN and the next level must be consulted.

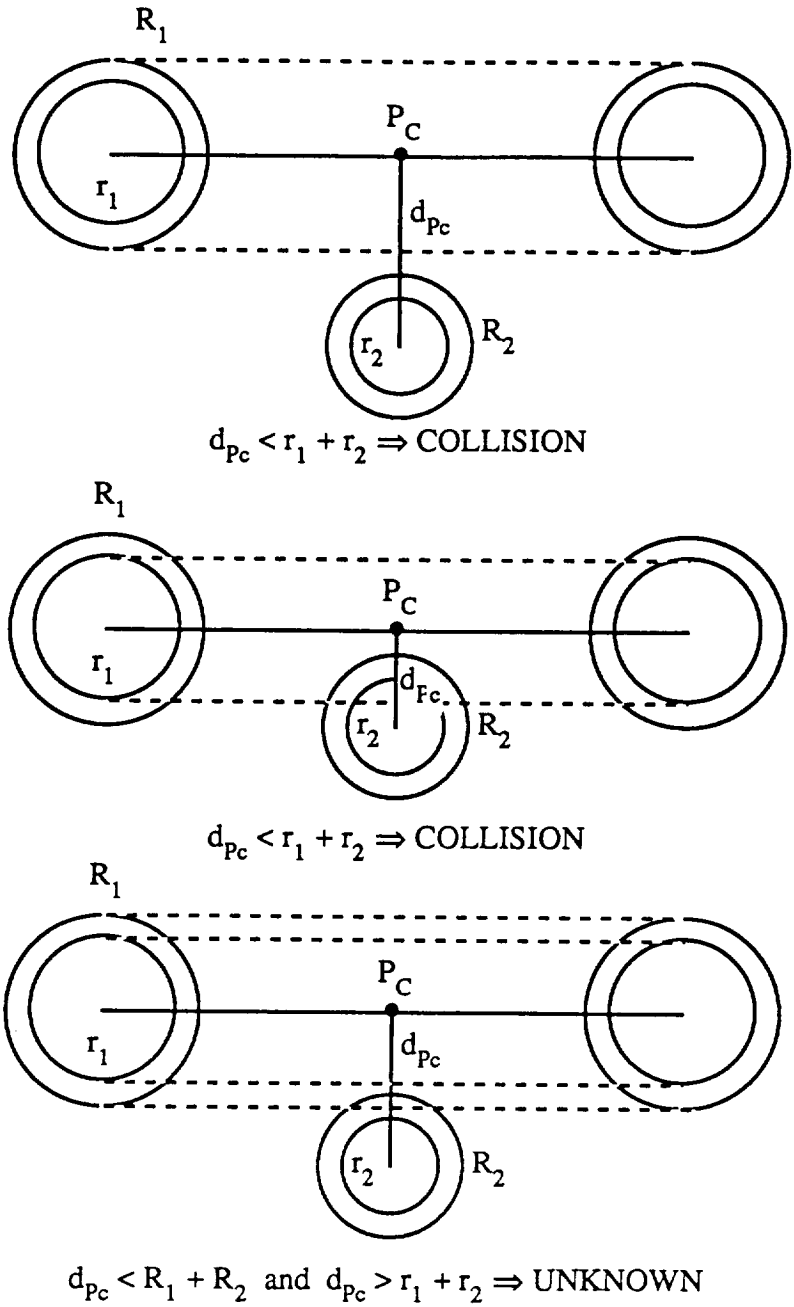


Fig. 6. Dynamic collisions - level 0, bounding sphere level possibilities.

Collisions at the face level are considerably simpler for line collisions than for point collisions. This is because the motion of the objects relative to each other prohibits an exact comparison of the

spherical sector associated with each face. Instead, a cumulative spherical sector encompassing the inner and outer radii of all spherical sectors within the range swept by the path is found for each object. The base angles for the range (α_{1base}, β_{1base}, α_{2base}, and β_{2base}) are determined by placing the moving object at Pc and proceeding in the same manner as with point collisions. Once the base range has been defined for each object, the spherical sectors are examined for intersection with it. The inner and outer radii (r_c and R_c) of the cumulative spherical sector are given by:

$$r_c = min_k\{r_k\} \text{ and } R_c = max_k\{R_k\} ,$$

where spherical sectors, S_k, are found to intersect with the base range. Collisions at this level are determined as follows:

$$\text{if } (r_{1c} + r_{2c} \geq d_{Pc}) \text{ then COLLISION;}$$
$$\text{if } (R_{1c} + R_{2c} < d_{Pc}) \text{ then NO COLLISION.}$$

If d_{Pc} is between the two sums, then the result in UNKNOWN and the next level must be used to determine a collision.

When a collision between two spherical sectors is undetermined at the facial level, the subface rectangular sectors are used to determine a collision. The points and angle ranges which define these spherical sectors are updated as they are needed to determine collisions only if the orientations of the objects have changed since the last collision. Determination of collisions proceeds in the same manner as the facial level: compare spherical sectors with base range, create cumulated spherical sectors from those which overlap, and compare radii of cumulated spherical sectors to determine a collision.

The method described provides collision detection information over the entire path to within the approximation error of the object representation. If more detailed information is required, incremental methods (via point collisions) or swept volume methods (via swept face collisions) such as described in the next section can be used.

IV. Collision Detection Scheme

The following section describes a collision detection scheme which uses the power of SSA techniques to perform complete detection between convex three-dimensional objects along a straight line path. The method uses line collisions to determine areas of the path requiring closer attention and point collisions on swept volumes within these areas.

A. Path Division

Line collisions are used as the first step in determining a collision between an object moving along a path and the objects in the environment. Areas for which there is no collision with any environment objects at the bounding sphere level identify portions of the path in which orientation changes can be made freely. The following is a technique which finds portions of the path for which a potential collision may occur.

As illustrated in Figure 7, the interval along the line where collisions could possibly occur has endpoints Q_1 and Q_2 given by:

$$d_Q = \sqrt{r_1^2 + r_2^2 - d_{Pc}^2} ,$$

$$Q_1 = Pc + d_Q u_l \text{ , and } Q_2 = Pc - d_Q u_l ,$$

where d_Q is half the length of the potential collision interval, r_1 and r_2 are the inner radii for the two bounding spheres, and u_l is the unit vector describing the direction of the straight line path.

The above method is also used to determine the area along the path for faces, of the moving object and faces of an environment object, which have been identified by the facial level as involved in a potential collision. This is accomplished by assigning r_1 and r_2 the values of the cumulative radii of the faces involved. These points are limited further by the endpoints of the path.

Once the potential collision points have been determined, a new volume is formed by sweeping a face of the moving object between the points and using SSA techniques to perform point collision detection. We call this technique swept face collision detection.

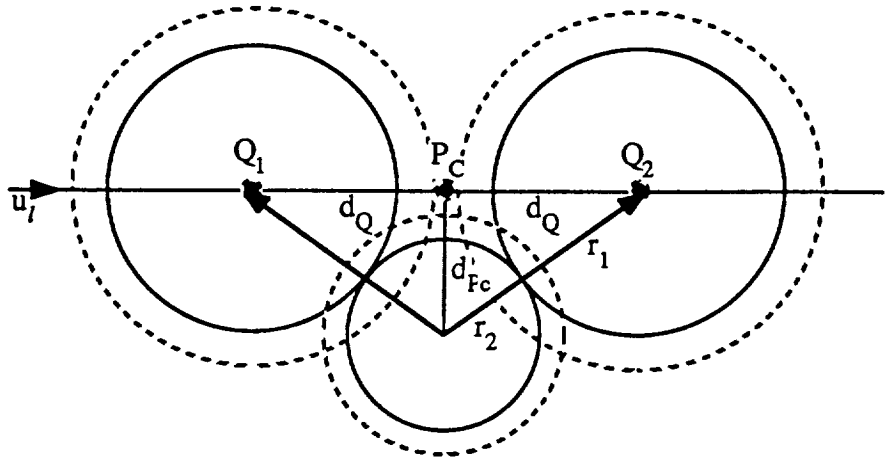


Fig. 7. Dynamic collisions - level 1, facial bounding sphere level: range of possible collision points.

B. Swept Face Collisions

SSA swept face collisions take over for the segments which are undetermined after the line detection strategy. At the conclusion of line collisions, we have a list of face pairs, which are involved in a potential collision, and a pair of endpoints which determine the range over which these points may collide. At this stage, point collisions could be used at intervals between the endpoints to determine a collision, however, it is difficult to determine what interval to use and guarantee a collision free path. Instead, we consider collisions between each face pair individually and use SSA techniques to determine if a collision occurs.

Swept face collision detection for a pair of faces starts at the point Pc where the two objects are closest along the path. If the outer radii of the segments do not intersect at this point, there is no collision regardless of the angles swept by the segments. If there is an overlap, a volume is formed by sweeping the payload face and comparing it to the stationary face of the obstacle. First the limits of the sweep which provide an area of potential collision are found. The points along the line at which the spheres first overlap are determined and the limits of the faces are examined to determine a segment of possible collision along the path. Portions of the path with coordinates outside the intersection are eliminated. The volume swept by the payload face is determined by forming a convex polygon from the positions of the vertices of the face at each of the endpoints of the path. If the direction of the sweep is perpendicular to the face normal, then the swept object degenerates to a swept face. This special case is handled by finding the boundaries of the face and comparing it to the stationary face of the obstacle.

In order to model the swept volume using SSA techniques, we must determine a reasonable center, such as the centroid of the volume, and create a representation as described in section II. An

SSA model based on this center is then created and a modified version of point collisions used between the swept volume and the object face to determine a collision. Because point collisions are used, the representation need not include the time consuming approximate subdivision of the faces. In addition, the planar face definitions are not created until they are required for the calculation of a planar intersection. Once the representation has been created, point collisions are used to determine an intersection between the swept volume of the payload face and the single face segment of the obstacle.

V. Path Planning

The rapidity with which collision detection can be done in three dimensions using SSA techniques makes it a practical method for use in planning collision free paths in operational space. A generate and test strategy requires many path iterations before a final path is determined, and therefore, needs a fairly rapid method of detecting path collisions. In addition to providing this, the SSA representation and our collision detection scheme give valuable information needed to make intelligent path modifications.

Some of this information can be derived directly from the collision detection process. For example, when detecting line collisions, the scheme identifies portions of the path where orientation changes can be made and path modifications may be required. The angular nature of the SSA representation also provides clues for possible intelligent modification of payload orientation, which aids path planning in tight spots.

Much of the research associated with path planning has been centered around the determination of a collision space of the robot. These methods give an exact determination of a path for a complex robot in a *static* environment. The cost of obtaining this exact path is in the extremely large amounts of time required to plan and, hence, this approach is unsuitable to dynamic environments. Operational space methods using a generate and test philosophy are not guaranteed to find a path in a very difficult situation. They also necessitate the additional iterative expense of exploring the collisions of each robot link after a path for the payload has been determined. However, we feel that our collision detection strategy will find a reasonable path through a realistic robot workcell environment in an acceptable amount of time. We expect it will be useful to solve problems in planning, with possible support from a C-space technique for "snapshots" of difficult situations, and in on-line path determination. Its ability to function in a dynamic 3-D environment may also be applicable to path planning situations with more than one moving object.

VI. Conclusions

In this paper, we have introduced a three-dimensional method of representing objects and illustrated its usefulness in quickly detecting collisions in a dynamic environment. Using the described SSA representation hierarchy, we are able to determine exactly if a given path for a convex three-dimensional object causes a collision with three-dimensional objects in its environment. Preliminary testing on a 68000-based computer indicates that the bounding sphere check takes less than 1 second per environment object along the path. The typical time to test a path is approximately 5 seconds for each object requiring processing beyond the bounding sphere level. The time for a worst case, a near miss, is less than 20 seconds; typically, the time for detecting an object collision is 2 seconds versus 10 seconds for non-colliding objects with indeterminate bounding spheres.

The SSA representation is significant because it is inherently three-dimensional and dynamic, making it well suited for modelling objects which are to be manipulated in a robot workcell. The SSA representation provides a useful, time conservative tool for use in a generate-and-test strategy for determining collision free paths. It provides a rapid method of collision detection in operational space and, thus, provides additional information which is useful for intelligent path modification.

ACKNOWLEDGEMENT

This research was motivated by work funded in part by the US-Spain Joint Committee for Scientific and Technological Cooperation (grant CA83-188).

REFERENCES

[1] V. Hayward, "Fast Collision Detection Scheme by Recursive Decomposition of a Manipulator Workspace," in *IEEE International Conference on Robotics and Automation*, 1986, pp. 1044-1049.

[2] T. Lozano-Perez, "Automatic Planning of Manipulator Transfer Movements," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 11, no. 10, pp. 681-698, October 1981.

[3] B. Faverjon, "Obstacle Avoidance Using an Octree in the Configuration Space of a Manipulator," in *IEEE International Conference on Robotics*, 1984, pp. 504-512.

[4] R.A. Brooks, "Planning Collision Free Motions for Pick and Place Operations," in *First International Symposium on Robotics Research*, 1984, pp. 5-37.

[5] L. Gouzenes, "Strategies for Solving Collision-free Trajectories Problems for, Mobile and Manipulator Robots," *The International Journal of Robotics Research*, vol. 3, no. 4, pp. 51-65, Winter 1984.

[6] M.C. Leu, "Developments in Robotics Software Systems," in *ASME Winter Annual Meeting*, pp. 123-132, 1983.

[7] A. Pennington, M.S. Bloor and M. Balila, "Geometric Modeling: A Contribution toward Intelligent Robots," in *13th International Symposium on Industrial Robots*, pp. 7.35-7.54, 1983.

[8] J.K. Meyers and G.J. Agin, "A Supervisory Collision-Avoidance System for Robot Controllers," in *ASME Winter Annual Meeting*, pp. 225-232, 1982.

[9] O. Khatib, "Real-Time Obstacle Avoidance for Manipulators and Mobile Robots," in *IEEE International Conference on Robotics and Automation*, pp. 500-505, 1985.

[10] V.J. Lumelsky, "Continuous Motion Planning in Unknown Environment for a 3D Cartesian Robot Arm," in *IEEE International Conference on Robotics and Automation*, pp. 1050-1055, 1986.